Grundlagen der Informatik III

Wintersemester 2013/2014

Prof. Dr.-Ing. Michael Goesele,
Simon Fuhrmann, Fabian Langguth

TECHNISCHE UNIVERSITÄT DARMSTADT

4. Praktikum

19.12.2013

Das Praktikum wird in Teams von drei Personen bearbeitet, die alle der selben Übungsgruppe angehören müssen. Vermerken Sie im Quellcode alle Teammitglieder mit Matrikelnummern. Die Lösung muss bis **spätestens 15.01.** im Moodle hochgeladen werden. Es genügt, wenn *ein* Teammitglied die Lösung abgibt. Den eigentlichen Testattermin vereinbaren Sie selbst mit Ihrem Tutor.

Geben Sie alle zum Kompilieren und Ausführen Ihres Programms benötigten Dateien in einem .zip- oder .tar.gz-Archiv ab. Ihre Abgabe muss mit den auf dem Aufgabenblatt angegebenen Anweisungen auf den RBG-Poolrechnern kompilier- und ausführbar sein, ansonsten wird sie mit 0 Punkten bewertet.

Wenn Sie dieses Praktikum erfolgreich bearbeiten und die Klausur bestehen, erhalten Sie **2 Punkte**. Da das Praktikum über die Weihnachtspause läuft, wird für das Bestehen nur ein sehr geringer Programmieraufwand benötigt. Um zusätzlich Bonuspunkte zu erreichen, werden jedoch weitere Aufgaben gestellt.

Sprechstunden für das Praktikum werden am 13. und 14.01.2014 von 16:30 bis 17:30 im C-Pool stattfinden.

Motivation

Im 3. Praktikum haben Sie sich bereits mit Beschleunigungsstrukturen und deren Implementierung beschäftigt. In diesem Praktikum geht es ausführlicher darum, das Rendern von 3D-Objekten zu implementieren. Dabei sollen parallele Methoden zur Beschleunigung der Berechnung genutzt werden.

Zur Verfügung steht Ihnen dabei bereits ein kleines Framework, das in der Lage ist, Dateien vom Typ .off, welche aus Dreiecken bestehende 3D-Szenerien enthalten, einzulesen und daraus eine Bounding-Volume-Hierarchy-Datenstruktur aufzubauen. Das Framework ist zum Großteil identisch zum Vorherigen Praktikum und Sie haben die Möglichkeit Ihre eigene BVH Struktur aus dem letzten Praktikum auch hier zu verwenden, in dem Sie einfach die entsprechenden Klassen austauschen. Weiterhin werden im Framework darin beretis Methoden zur Verfügung gestellt, mit denen Rendering über einfaches Raycasting möglich ist. Zuletzt können auf diese Weise gerenderte Dateien auch bereits als .pgm-Bilddatei gespeichert werden.

Konzepte

Raycasting

Raycasting ist eine einfachere Variante des heutzutage recht gebräuchlichen Renderingverfahrens des Ractracings. Vereinfacht gesagt, geht man beim Raycasting von einem Augpunkt oder einer Kamera als Betrachter einer Szenerie aus und besitzt zwischen diesem und der Szenerie eine Bildfläche, auf welche die im Raum verteilten Objekte abgebildet werden sollen. Die Vorgehensweise des Raycasting-Algorithmus ist nun, dass durch jeden Bildpunkt der Projektionsfläche ein (später mehrere) Sichtstrahl, ein so genannter „Ray“ geschickt wird. Dieser enthält Informationen darüber, von welchem Bildpunkt er in welche Richtung ausgeschickt wurde und liefert beim Auftreffen auf ein Objekt im Raum an besagten Bildpunkt die Farbinformationen des getroffenen Objektes zurück. Trifft der Strahl nach Zurücklegen einer initial festgelegten Entfernung auf kein Objekt, wird der Bildpunkt, von dem der Strahl ausgesandt wurde, auf eine Default-Farbe (schwarz) gesetzt.

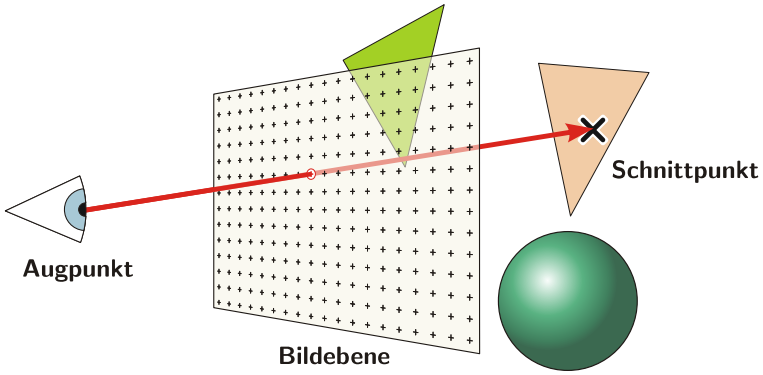

Abbildung 1: Ein Strahl wird vom Augpunkt durch einen Pixel geschossen, trifft auf ein Objekt und färbt daher den Pixel, durch den er geschossen wurde, in der Farbe des Objekts. Quelle: `http://de.wikipedia.org/w/index.php?title=Datei:Raytracing.svg&filetimestamp=20070124164523` (User Phrood)

Supersampling

Als Supersampling bezeichnet man ein Verfahren zur Verbesserung von Ergebnissen zum Beispiel von Bildberechnungsverfahren. Abstrakt gesprochen wird hierbei die (Bild-)Qualität durch so genannte „Überabtastung“ erhöht, einer der sichtbaren Effekte ist hierbei ist „Anti-Aliasing“, im Fall der Computergrafik spricht man auch oft vereinfachend von Kantenglättung. Konkret wird Supersampling so umgesetzt, dass man die Bildfläche feiner unterteilt, als sie eigentlich ist, also durch jeden Pixel beispielsweise vier oder 16 Strahlen mit leicht unterschiedlichen Winkeln schickt und deren zurückgelieferte Werte dann anteilig (also geviertelt bzw. gesechzehntelt) auf den Bildpunkt aufaddiert.

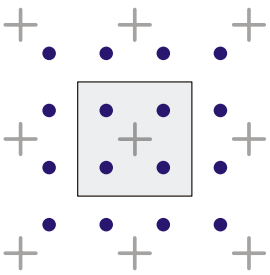

Abbildung 2: Statt nur einem Strahl pro Pixel (die Kreuze) werden in diesem Beispiel pro Pixel 4 Strahlen (die blauen Punkte) mit leicht abweichenden Richtungen vom Augpunkt durch die Bildebene in die Szenerie geschossen und ihre Ergebnisse gemittelt. Quelle: `http://upload.wikimedia.org/wikipedia/commons/3/39/Supersampling_-_Uniform.svg` (Public Domain)

Shading

Eine mit den bisher vorgestellten Techniken gerenderte Szenerie wird trotz bildverbessernden Samplings sehr unnatürlich und vor allem „platt“ aussehen. Das liegt daran, dass Raycasting an sich zunächst keine besonderen Beleuchtungs- bzw. Schattierungsmethoden („Shading“) implementiert. Jedem Dreieck wird abhängig von seinem Winkel zur Lichtquelle genau eine Farbe zugeordnet. Man spricht in diesem Fall von „Flat Shading“. Man kann deutlich bessere Ergebnisse erzielen, wenn man die Normale auf dem Dreieck anhand seiner Eckpunkte interpoliert und die Farbe dann für jeden Ray, abhängig von seinem genauen Schnittpunkt auf dem Dreieck, einzeln berechnet. Dies ist jedoch immer noch eine sehr lokale Methode.

Eine auf dem Raycasting-Prinzip aufbauende globalere Shading-Methode ist das so genannte „Ambient Occlusion Shading“. Dabei werden, wenn ein Ray auf ein Objekt trifft, nochmals einige Strahlen generiert, welche dann von der getroffenen Fläche halbkugelförmig in Richtung der Normalen der Fläche (der senkrecht auf der Fläche

Abbildung 3: Fertig gerendertes Bild mit 1^2, 2^2 und 3^2 Supersamples. Feine Strukturen können durch mehr Supersamples deutlich besser gerendert werden.

stehenden Gerade) ausgeschickt werden. Diese zusätzlich ausgeschickten Strahlen lediglich die wahrgenommene Helligkeit der Fläche, von der sie ausgesandt wurden.

Trifft also ein Ambient Occlusion Ray auf ein anderes Objekt, testet man, wie weit dieses Objekt vom Ursprung des Strahls entfernt ist. Liegt der Abstand zum Schnittpunkt über einem Vorgegebenen Schwellwert (`MaxDistance`) wird davon ausgegangen, dass das getroffene Objekt die Beleuchtung des ursprünglichen Schnittpunktes nicht beeinflusst. Liegt der Abstand jedoch unterhalb von `MaxDistance`, wird davon ausgegangen, dass jenes Objekt ein wenig Schatten auf das ursprüngliche Objekt wirft, von dem die Strahlen ausgesandt wurden. Der entgültige Ambient Occlusion Faktor ist schließlich einfach der Anteil der Strahlen, die auf ein schattenwerfendes Objekt treffen. Werden zum Beispiel 16 Strahlen von einer Oberfläche ausgeschickt, und 7 davon treffen ein Objekt innerhalb von `MaxDistance`, so ist der Ambient Occlusion Faktor 7/16. Dieser Faktor wird einfach auf den ursprünglichen Wert des Shadings multipliziert und somit der Schattenwurf simuliert.

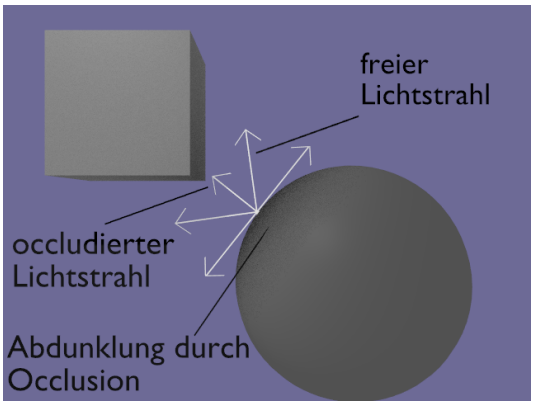

Abbildung 4: Von einem Auftreffpunkt eines Sichtstrahls werden Ambient Occlusion Rays in Richtung der Oberflächennormalen ausgesandt. Manche werden occludiert, manche nicht. Nur die unoccludierten tragen zur Helligkeit des zugehoerigen Pixels bei. Quelle: `http://upload.wikimedia.org/wikibooks/de/3/3a/Ao_erklaerung.png` (Public Domain)

Aufgabenstellung

Ihre Aufgabe ist die Implementierung von Supersampling und Ambient Occlusion. Dazu sollen Sie zum Bestehen des Praktikums:

- in die vorgegebene `RayTracer::trace` Methode ändern und Supersampling implementieren um das resultierende Bild zu entstören. Bisher erzeugt die Methode genau einen Strahl für jeden Bildpunkt. Ihre Lösung

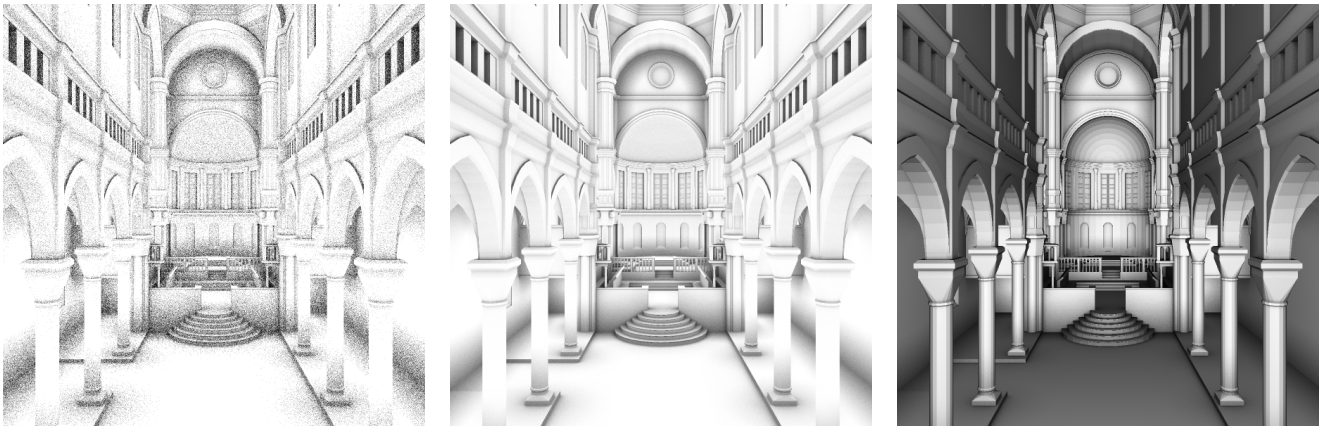

Abbildung 5: Links: Nur Ambient Occlusion mit 10 Samples. Mitte: Nur Ambient Occlusion mit 1000 Samples. Rechts: Normales Shading mit 2^2 Multisamples und Ambient Occlusion mit 1000 Samples.

soll schließlich n^2 Strahlen pro Pixel mit einer entsprechenden uniformen Verteilung erzeugen und die Ergebnisse aller Strahlen mitteln.

- die Methode `RayTracer::ambientOcclusion` implementieren und mit Hilfe der Klasse `HemisphereSampler` den Ambient Occlusion Faktor für jeden erzeugten Ray, der auf ein Objekt trifft berechnen.

Bonus

Bonuspunkte für dieses Praktikum können Sie wie folgt erreichen, bitte achten Sie dazu auch auf die Hinweise:

- 1 Punkt, falls sie eine neue `Ambient Occlusion Intersect` Methode in die BVH Struktur integrieren, die das Auswerten der Ambient Occlusion beschleunigt.
- 1 Punkt, falls Sie eine der Intersect Methoden (Triangle oder AABB) mittels SSE beschleunigen.

Hinweise

Ambient Occlusion

Da Ambient Occlusion nur innerhalb einer `MaxDistance` eine Rolle spielt, macht es Sinn gewisse teile der BVH nicht zu traversieren, wenn bereits frühzeitig klar ist, dass die `MaxDistance` auf jeden Fall überschritten wird. Konkret kann man das Traversieren der BVH einfach Abbrechen, wenn der Schnittpunkt des Strahls mit einer der AABBs bereits außerhalb der `MaxDistance` liegt. Um dies zu Implementieren müssen Sie zunächst die Intersect Methode der AABB ändern, so dass sie auch die Distanz des Strahls zur AABB zurück bekommen. Es sind auch weitere Mechanismen denkbar. Zum Beispiel kann man immer zuerst in den Ast der BVH absteigen, der näher zum Strahlursprung ist.

Wettbewerb

Zusätzlich wird es einen kleinen Wettbewerb geben, bei dem wir die schnellste aller teilnehmenden Lösungen in einer der kommenden Vorlesungen auszeichnen. Wir werden dabei ausschließlich die Zeit, die zum Rendern benötigt wird, messen (also nicht die Zeit, die zum BVH bauen benötigt wird). Entsprechend gute Lösungen aus dem letzten Praktikum können hier also auch vorteilhaft verwendet werden. Der Wettbewerb ist vollständig unabhängig vom Bestehen des Praktikums. Wir werden eine explizite zusätzliche Abgabe im Moodle einrichten. Denken Sie also, falls Sie teilnehmen möchten, daran ihre reguläre Lösung auch in der dafür vorgesehenen Abgabe abzulegen und die Wettbewerbslösung gesondert abzugeben.

Hier noch einige Randbedingungen:

- keine Einschränkungen für die konkrete Implementierung
- Das Programm soll direkt lauffähig sein d.h. keine weiteren Installationen erfordern.
- Ihre Lösung muss selbst entwickelt sein (wir kennen die Systeme im Internet auch)
- Kriterien für die schnellste/beste Lösung:
  - Geschwindigkeit
  - Skalierbarkeit (Szenengröße, Threadzahl, Supersampling, AO, ...)
  - NICHT: die meisten Features!!!
- Wir behalten uns vor, die Kriterien je nach Verlauf des Wettbewerbs anzupassen.
- Der Rechtsweg ist ausgeschlossen. Alle Entscheidungen der Preiskommission sind endgültig und unanfechtbar.

Viel Erfolg!